통합 인증 시스템 설계 및 구현

연세대학교 전기·전자공학과
정연식, 송홍엽
Contents

- Introduction
- Previous Works
- Design and Implementation of Public–Key Infrastructure
- Design and Implementation of Single Sign–On
- Conclusion
Introduction

Multi-Server Environment

- Problems of multiple passwords
- Problems of implementation of challenge–response protocol
- Single Sign-On

Single Sign-On

- Integration of authentication schemes inside domain of service
- A user logs in once using a single password, and gets authenticated access to all servers in the Intranet
- Without sending any passwords over the network
Previous Works: Kerberos

Kerberos System

- Based on secret-key cryptosystem
- Based on Needham-Schroeder’s third-party protocol

Diagram:

- User
- Client
- Authentication Server
- Ticket-granting Server
- Application Server
- Database

Flow:

- User requests TGT from Authentication Server
- Authentication Server sends TGT + Session-key to Ticket-granting Server
- Ticket-granting Server sends SGT + Session-key to Application Server
- Application Server requests service
- Application Server gets TGT and SGT from Ticket-granting Server

Requests:

- Once per user log-in
- Once per type of service
- Once per service session

Note:

- Kerberos System is based on secret-key cryptosystem and Needham-Schroeder’s third-party protocol.
Netscape’s SuiteSpot Server

- Based on digital signature
- Based on Secure Socket Layer (SSL)

1. User enters private-key password.
2. Client retrieves private-key and generates the digital signature.
3. Client sends the certificate and the digital signature over the SSL connection.
4. Server authenticates user’s identity.
5. Server checks whether certificate is in LDAP entry for user.
6. Server authorizes access for user.
The Proposed Single Sign-On

- Based on Public-Key Infrastructure
- Use of challenge–response protocol

Diagram:
- User
- Client
- Challenge-response protocol
- Application Server 1
- Authentication Server
- LDAP Server
- Application Server 2
- CA Server
Contents

- Introduction
- Previous Works
  ✔ Design and Implementation of Public-Key Infrastructure
  ✔ Design and Implementation of Single Sign-On
- Conclusion
Public-Key Infrastructure

**Concept**
- Enables the use of public-key encryption and digital signature in a consistent manner
- Certificate management infrastructure
  - Issues and provides access to public-key certificates
- Network of certificate authorities

**Components**
- Certificate Authority (CA) server
- Directory server
- Client modules
Generation of Certificate

User

Generate a public/private-key pair and use PKCS #5 to encrypt the private-key

Client

Process PKCS #10 to send certification request.

CA Server

Sign and Verify data using PKCS #7.

Application Server

Request certificates and CRLs.

LDAP Server

Publish certificates and CRLs.

Another Intranet

Cross-certification

CA Server
CA Server

Functions
- Generates certificates
- Registers the certificates into the directory server
- Issues certificates
  - Using PIN
- Revokes certificates
  - Simply deletes a user entry and its attribute from the directory
  - Does not need the use of certificate revocation list (CRL)

Implementation
- Apache web server
- CGI programming
### X.509 Certificate Format

- **Version**
  - ver.1
- **Signature algorithm identifier**
  - MD5 with RSA encryption
- **Issuer name**
  - Root CA (Yonsei CA)
- **Period of validity: Not before**
  - Present date
- **Period of validity: Not after**
  - Present date + 1 year
- **Subject’s public-key info**
  - RSA algorithm

---

**Version**

<table>
<thead>
<tr>
<th>Certificate serial number</th>
<th>Algorithm Parameters</th>
<th>Issuer name</th>
<th>Period of validity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Not before</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Not after</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Subject name</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Algoths Parameters</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Key</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Issuer unique identifier</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Subject unique identifier</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Extensions</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Algorithms Parameters</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Encrypted</td>
</tr>
</tbody>
</table>
Directory Server

Functions
- Stores the certificates and CRLs
- Makes all stored certificates available on request

Lightweight Directory Access Protocol (LDAP)
- Simplified DAP (X.500)
- Runs directly over TCP or other reliable transport layer

Implementation
- OpenLDAP release distribution
  - Stand-alone LDAP daemon: SLAPD server
- Binary to ASCII conversion
  - Base-64 encoding/decoding algorithm
Client Module

Functions
- Generates public/private-key pairs
  - RSA key pair
  - Size of modulus: 1024bit
- Encrypts the private-key and saves into the local machine
  - Password-based encryption
- Generates certification requests
  - PKCS #10
- Communicates with the CA server to send certification request

Implementation
- Visual Basic Script language
- Microsoft ActiveX control programming
- Socket programming
언론사 신청

언론사 신청방식입니다. 모든 정보는 영문으로 입력해야 하지 않습니다.

업무상에 E-Mail(이메일)은 네트워크상으로 전송되지 않으니, 

- 이름
- 국가
- 시 도
- 기관(회사)
- 전화번호

내부번호는 현재까지 입력하실 수 오직 Pass-Pass만 사용할 수 있습니다.(문제 23. 인물 187)

내부번호
내부번호 확인

언론사 발급 및 변경, 재소를 위한 PR 번호를 받아야 하는 E-Mail 주소를 작성해 주십시오.

E-Mail

[전달] [재입력]

언론사 신청이 처리되는 때로 E-Mail을 통해 PR 번호를 보내드립니다.

언론사 활동해지시에 PR 번호를 이용하여 언론사를 해지해야 사용하실 수 있습니다.
Contents

- Introduction
- Previous Works
- Design and Implementation of Public-Key Infrastructure
  - Design and Implementation of Single Sign-On
- Conclusion
The Proposed Single Sign-On

Components

- Authentication server
- Application servers
- Client modules

User

Step 1

Step 2

User

Application Server

Challenge-response protocol

Step 3

Step 4

Step 5

Step 6

LDAP Server

Authentication Server

Application Server

Client

Application Server
Requirements Capture

Message Sequence Chart

- Guide for designing the system
- Visualization of the system runs

<table>
<thead>
<tr>
<th>Messages</th>
<th>Subjects</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>service_req</td>
<td>Client (User)</td>
<td>URL, userEmail</td>
</tr>
<tr>
<td>service_resp</td>
<td>↔ Application Server</td>
<td>flag</td>
</tr>
<tr>
<td>auth_req</td>
<td>Application Server</td>
<td>serverName, userEmail, randStr</td>
</tr>
<tr>
<td>auth_resp</td>
<td>↔ Authentication Server</td>
<td>userEmail, ticket</td>
</tr>
<tr>
<td>password_req</td>
<td>Authentication Server</td>
<td>–</td>
</tr>
<tr>
<td>password_resp</td>
<td>↔ Client (User)</td>
<td>flag</td>
</tr>
<tr>
<td>cert_req</td>
<td>Authentication Server</td>
<td>–</td>
</tr>
<tr>
<td>cert_resp</td>
<td>↔ LDAP Server</td>
<td>certificate</td>
</tr>
<tr>
<td>signature_req</td>
<td>Authentication Server</td>
<td>randNum</td>
</tr>
<tr>
<td>signature_resp</td>
<td>↔ Client (User)</td>
<td>signedData</td>
</tr>
</tbody>
</table>
Scenario 1

First login: General case
In case a user enters invalid password
Scenario 3

In case a user accesses the same application server from the same client within timer expiration
In case a user accesses another application server from the same client within timer expiration
In case an opponent accesses the application server from the another client within timer expiration.
**Scenario 6**

**randStr & ticket**
- Application server can be accessed through two URIs
- To prevent an opponent from directly accessing an application server without authentication protocol
- Can prevent server-spoofing attack
- ticket: randStr encrypted by authentication server

In case an opponent directly accesses an application server without authentication process

Client (User) <-> Application Server <-> Authentication Server <-> LDAP Server

service_req

service_resp (F)
In case a user sends multiple service_req messages
Authentication Server

Functions

- Challenge–response protocol
- Generates the ticket
  - DES–CBC with all 0’s IV
  - To transmit as CGI parameter
    - Base-64 encoding algorithm
    - Additional encoding: + → −, / → @, = → :
- Control of the timer($T_{\text{Auth}}$)
  - Implemented by checking the log-in time in the database
  - 1 hour

Implementation

- Apache web server
- CGI programming
- Database management system
  - MySQL database
Application Server

Functions

- Verifies the ticket
  - DES-CBC with all 0’s IV
  - Base-64 decoding algorithm
- Control of the timer ($T_{App}$)
  - Maximum Segment Lifetime (MSL)
  - 30 seconds

Implementation

- Apache web server
- CGI programming
- Redirection
  - Location header of HTTP
Client Module

Functions

- Decrypts the private-key
  - Password-based decryption
- Challenge-response protocol
  - RSA signature algorithm with MD5 hash function
  - Size of modulus: 1024bit

Implementation

- Visual Basic Script language
- Microsoft ActiveX control programming
- Socket programming
Conclusion

Advantages

- Has the mechanism that a user directly connects the application server he wants to access
- Independent of lower layer protocol implementations
- Easy to implement and use in the Intranet

Future Research

- Development of SSO for other application protocols
  - Current: HTTP
  - Future: Telnet, customized application protocols
- Extension to multiple-CA environment
- Use of smart card